WWW.SCIAM-MAGAZINE.NAROD.RU

О фрактальных горах, графтальных растениях и других графических чудесах фирмы Pixar

А. К. ДЬЮДНИ

Я ПРЕДСТАВЛЯЮ себе появление первого полнометражного фильма, созданного компьютером. Год 1991. По лестнице спускаюсь в партер зрительного зала с большой сумкой, набитой пакетиками с синтетическими кукурузными хлопьями и бутылками с прохладительными напитками, уже не содержащими обычных натуральных ингредиентов. Медленно гаснет свет в зале, занавес раздвигается и серебряный экран оживает сценой из фильма, созданного по мотивам трилогии Дж. Толкейна «Властелин колец». Фродо Хоббит прогуливается по узкой горной долине. Вдали острые заснеженные горные вершины упираются в самое небо. На переднем плане в ярких солнечных лучах поблескивают экзотические деревья и диковинные травы. Эту сцену сменяет другая — волшебник смотрит в хрустальный шар. В центре шара появляется крепость, ее стены объяты пламенем.

Сейчас трудно сказать, насколько убедительными будут казаться движения и голос Фродо в таком фильме, но я уверен, что горы, растения, хрустальный шар и пламя получатся превосходно. Этим успехом фильм будет в значительной мере обязан новому программному обеспечению и аппаратуре, созданным фирмой Pixar (в прошлом лаборатория компьютерной графики Lucasfilm). Посетив эту удивительную мекку компьютерной графики в Сан-Рафаэле (шт. Калифорния), я готов поделиться с читателями всеми секретами построения изображений гор и растений. Любой, кто располагает домашним компьютером, сможет теперь самостоятельно создавать убедительные изображения подобных объектов. Правда, объем этой статьи не позволит подробно рассмотреть процедуру формирования изображения хрустального шара и пламени, но тем не менее я постараюсь осветить в общих чертах принципы, на основе которых создаются их изображения.

В нашем гипотетическом фильме камера может как бы приближать к нам заснеженные вершины гор, возвышающиеся за спиной Фродо. Более устрашающие горные хребты, наверное, трудно себе представить. Каждая крупная вершина состоит из отдельных более мелких пиков, а те в свою очередь состоят из еще более мелких и т. д. Здесь перед нами раскрывается бесконечная регрессия пиков. Даже Орк (из упомянутого фильма) — зигзагообразный зверь с толстокожими жесткими ступнями чувствовал бы себя неуютно на подобных склонах с острыми выступами.

Создать изображение такого горного хребта в принципе не сложно. Предположим для простоты, что наша территория имеет треугольную форму. Исходный большой треугольник можно поделить на четыре треугольника меньших размеров, нанеся точки на середине каждой стороны треугольника и соединив эти точки между собой. Каждый из вновь образованных треугольников поделим аналогичным образом. Этот процесс можно продолжать до тех пор, пока не будут достигнуты пределы оптического разрешения или машинного времени компьютера. Полученную в итоге довольно скучноватую решетку треугольников можно несколько оживить, прибегнув к вертикальным вариациям. Каждый раз, найдя среднюю точку на стороне треугольника, ее можно поднять или опустить на некоторое случайным образом выбранное расстояние. Вертикальные перемещения, которые, вообще говоря, должны уменьшаться по мере того, как треугольники становятся все меньше, преобразуют треугольники в горный ландшафт с пиками и расщелинами (см. верхний рисунок на с. 106).

Почему благодаря этому методу удается построить достаточно естественно выглядящие горы? Отчасти, наверное, потому, что описанный процесс позволяет построить объект, обладающий фрактальными свойствами: при нарастающем увеличении объекта видны все более и более мелкие его детали. Бенуа Мандельброт, неутомимый «фракталист», работающий в исследовательском центре Томаса Уотсона фирмы IBM в Йорктаун-Хайтсе (шт. Нью-Йорк), в качестве примера, объясняющего свойства таких объектов, использует береговую линию. Допустим, что вы должны измерить длину береговой линии Франции при помощи мерного шеста длиной в один километр. Перебрасывая мерный шест с одного конца на другой, пройдем вдоль всей береговой линии и подсчитаем число отмеренных километров. Конечно, при этом мы не учтем множество мелких заливчиков и мысов, поэтому наш окончательный результат будет не очень точным. Повторив эксперимент с метровой палочкой, получим более точный и больший по величине результат. Но и в этом случае бесчисленное множество мельчайших выступов и впадин окажется неучтенным. Несомненно, сантиметровая палочка еще больше уточнит результат измерения.

Общая закономерность, таким образом, заключается в том, что измеренная длина береговой линии увеличивается с уменьшением длины измерительного прибора. Отношение измеренной длины контура к длине мерной палочки* — это особое число. Оно называется фрактальной размерностью. Фрактальная размерность отличается от обычного понятия размерности тем, что она обычно выражается дробным числом. Береговая линия, например, может иметь фрактальную размерность, равную 3/2. Такой объект можно представлять себе как нечто промежуточное между одномерным объектом (прямой линией) и двумерным объектом (плоскостью). Если береговая линия относительно прямолинейная, ее фрактальная размерность будет близка к 1, если же береговая линия сильно изрезана, ее фрактальная размерность будет приближаться к 2. Такая линия как бы стремится заполнить двумерное плоское пространство.

Фрактальная модель природного объекта влечет за собой явление бесконечной регрессии формы. С точки зрения компьютерной графики бесконечная регрессия — это в общем-то высокие материи. Достаточно того, чтобы ландшафт имел детали при всех возможных уровнях увеличения. В пределах разрешения экрана горную поверхность можно изобразить с достаточно мелкими деталями, которые можно построить, пользуясь описанным выше процессом последовательного деления треугольников. Хотя полное описание алгоритма, используемого для изображения горно-

* Очевидно, автор имеет в виду отношение истинной длины контура к измеренной при помощи мерной палочки. — *Прим. перев.*

WWW.SCIAM-MAGAZINE.NAROD.RU

го ландшафта, из-за его сложности привести в данной статье не представляется возможным, существует простая программа MOUNTAIN (гора), которая изображает пик Мандельброта в поперечном сечении. Программа MOUNTAIN хорошо иллюстрирует основную идею перемещения точек деления вдоль вертикальной оси. Художник-фракталист начинает рисунок с одного горизонтального отрезка прямой. Определив среднюю точку отрезка, он отклоняет ее вверх или вниз на случайно выбираемое расстояние. Каждый из двух новых отрезков снова делится пополам, и средняя точка подвергается перемещению. Этот процесс можно продолжать неопределенно долго, так же как и деление треугольников.

В программе MOUNTAIN имеются два массива: *points* (точки) и *lines* (линии). В них хранится информация о горном профиле. Каждый массив содержит две колонки и некоторое количество строк (скажем, 2048), достаточное, чтобы достичь разрешения экрана. В двух столбцах массива *points* хранятся координаты, а в двух столбцах массива *lines* — индексы. Каждая линия задается двумя точками в массиве *points* — значениями координат начальной и конечной точек линии. Чтобы пользователь мог визуально следить за интересным процессом последовательных делений отрезков и образования контуров горного хребта из исходного ничем не примечательного многоугольника, программа останавливается на каждом новом поколении точек и ожидает, какое решение он примет. При завершении главного и единственного цикла программа запрашивает пользователя, нужна ли еще одна итерация. Если он отвечает «да», управление опять передается на начало программы.

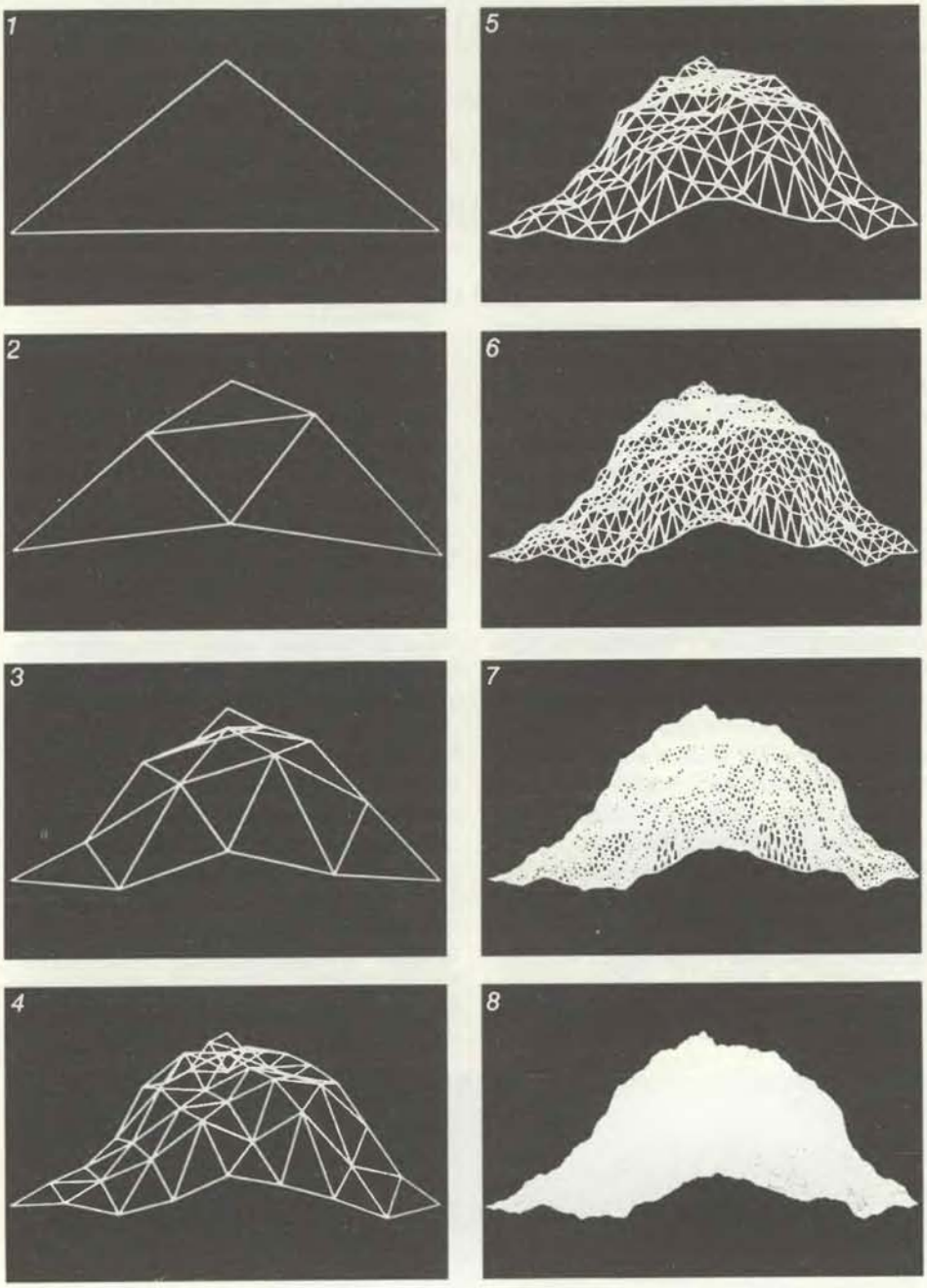
В теле главного цикла текущие множества точек и линий преобразуются в новые множества, вдвое более многочисленные. С этой целью программа просматривает массив *lines* строку за строкой, находит индексы соответствующих точек и извлекает их координаты из массива *points*. По известным координатам конечных точек отрезка прямой вычисляются координаты средней точки отрезка, и координата по оси *y* изменяется случайным образом. Приведенная ниже формальная запись алгоритма может послужить основой для создания программы. В этой записи переменные *j* и *k* указывают на строки массивов *points* и *lines*, которые в данный момент заполняются значениями, полученными в результате последнего деления. В переменных *pts* и *lns* хранится количество точек и линий, составляющих профиль горы перед последним входом в тело главного цикла. Первоначально *j* равно *pts*, а *k* равно *lns*. Индекс *i* пробегает значения от 1 до *lns*.

```
j ← j + 1
k ← k + 1
a ← lines(i,1)
b ← lines(i,2)
x1 ← points(a,1)
y1 ← points(a,2)
x2 ← points(b,1)
y2 ← points(b,2)
points(j,1) ← (x1 + x2)/2
points(j,2) ← ( y 1 + y 2 ) /2 +
              + random(range)
lines(i,2) ← j
lines(k,1) ← j
lines(k,2) ← b
```

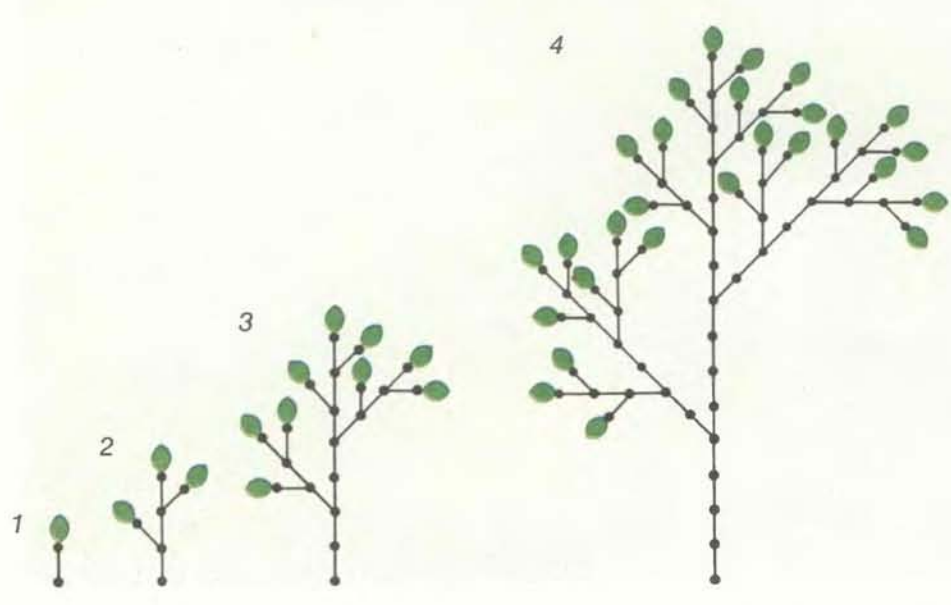
Эта часть программы MOUNTAIN почти не нуждается в дополнительных пояснениях. После того как вычислены координаты *j*-й точки значение индекса *j* запоминается как указатель на вторую точку *i*-й линии и на первую точку *k*-й линии. Первая точка *i*-й линии остается той же, что и прежде, а вторая точка *k*-й линии идентична исходной второй точке *i*-й линии; эта точка имеет индекс *b*. Когда тело цикла полностью выполнено, значения переменных *pts* и *lns* должны быть установлены равными последним значениям *j* и *k* соответственно. Значение переменной *range* первоначально устанавливается пользователем как максимально возможная величина случайного вертикального

Фрактальные горы, построенные компьютером, имитируют природные

WWW.SCIAM-MAGAZINE.NAROD.RU



Гора получается делением треугольника



Четыре поколения графтального растения

смещения для точек деления. После каждого выполнения тела цикла значение этой переменной должно быть уменьшено вдвое, чтобы сохранить пропорцию между масштабом случайных вариаций и размером форм, подвергающихся этим вариациям. Функция *random* (*range*) выражает выбор случайного числа в диапазоне от 0 до текущего значения переменной *range*.

Если горы за спиной у Фродо выглядят впечатляющими, то окружающие его со всех сторон растения и деревья никак им не уступают. Они выглядят и реалистично, и в то же время как-то сказочно-причудливо. Реалистичными они выглядят вследствие того, что ветвятся аналогично настоящим растениям, а необычными они кажутся потому, что не принадлежат знакомым видам. В распоряжении графического дизайнера так много различных параметров, что он не может устоять перед искушением создать нечто новое.

Эти новые «виды» растений можно, пожалуй, назвать «графтальными», потому что они основаны на графах и имеют неявно выраженную фрактальную природу. Говоря о неявной фрактальной природе, я имею в виду, что правила построения топологии растений могли бы применяться (но не применяются) вплоть до предела, накладываемого разрешением экрана. Короче говоря, ветви не разветвляются до бесконечности. После того как граф, лежащий в основе растений, построен, из него можно вывести мириады убедительно выглядящих видов, интерпретируя этот граф по-разному в отношении размеров, цвета, густоты, фактуры и т. д.

Графы, определяющие топологию того или иного растения, генерируются при помощи *L*-систем, образующих класс грамматик, которые были введены датским биологом и математиком Аристидом Линденмеером в 1968 г. *L*-система по существу представляет собой набор правил, позволяющих выводить новые цепочки символов из старых. Эти правила задают способ подстановки последовательностей символов на место одного исходного символа. Пользуясь, например, алфавитом из символов 0,1, а также скобками [и], можно построить множество сложных ботанических форм, если воспользоваться следующими правилами:

0 → 1[0]1[0]0
1 → 11
[→ [
] →]

Чтобы посмотреть, как работают эти правила, допустим, что у нас есть

WWW.SCIAM-MAGAZINE.NARODRU

одна исходная цепочка символов, состоящая из единственного символа 0. Последовательно заменяя его и каждый встречающийся далее символ соответствующей ему цепочкой по приведенным правилам, получим такую последовательность символьных цепочек:

```
0
1[0]1[0]0
11[1[0]1[0]0]11[1[0]1[0]0]1[0]1[0]0
```

Подобные цепочки можно преобразовать в древовидные графы, если каждое число (0 или 1) представить в виде отрезка прямой, а каждую скобку — точкой ветвления. Отрезки, соответствующие 0 и 1, имеют одинаковую длину, но обычно отличаются тем, что ветки типа 1 остаются голыми, а на конце каждой ветки типа 0 находится лист.

Например, стебель (или ствол) цепочки 1[0]1[0]0 состоит из трех символов, не заключенных в скобки. Отрезок типа 1 продолжается другим таким же отрезком и завершается отрезком типа 0. Две ветки, каждая из которых состоит из единственного отрезка 0, отделяются от стебля, описанного этой формулой. Первая ветвь отходит от конца первого отрезка, а вторая — от конца второго отрезка. Прежде чем мы перейдем к рассмотрению нижнего рисунка на с. 106, читатели могут попрактиковаться в построении первых нескольких поколений структуры. Для достижения большей реалистичности в описанную модель можно ввести некоторые дополнительные свойства, например наложить условие, согласно которому для каждого данного стебля (независимо от того, является ли он стволом или веткой) вторичные ветви должны отходить поочередно влево и вправо. Читатели, желающие составить собственные графтальные программы, могут ограничиться этой упрощенной моделью растений. При помощи рассмотренной грамматики профессионалы, работающие в фирме Pixar, получают довольно красивые растения, подобные тем, что показаны на верхнем рисунке справа.

Состоящая из двух частей программа под названием PLANT (растение) строит n-ю цепочку в рассмотренной выше последовательности и затем трансформирует ее в изображение в виде контурного рисунка. В первой части программы генерируемые ею цепочки запоминаются в двух символьных массивах *string A* и *string B*. Каждое поколение растений занимает один из этих массивов поочередно: данное поколение в одном массиве выводится из предыдущего поколения, которое содержится в другом массиве. Хранить в этих массивах символы не обязательно. С тем же успехом можно воспользоваться цифрами 0, 1, 2 и 3 при условии, что программа будет правильно выполнять подстановки.

Правила L-системы встроены в программе в условные операторы. Например, приведенная ниже процедура, записанная в алгоритмическом виде, может служить для преобразования 0 на i-позиции цепочки символов *string A* в девять новых символов, которые будут записаны в цепочку *string B*:

```
если stringA(i) = 0, тогда
    stringB(j) ← 1
    stringB(j + 1) ← 2
    stringB(j + 2) ← 0
    stringB(j + 3) ← 3
    stringB(j + 4) ← 1
    stringB(j + 5) ← 2
    stringB(j + 6) ← 0
    stringB(j + 7) ← 3
    stringB(j + 8) ← 0
    j ← j + 9
```

Здесь 0 и 1 обозначают сами себя, а 2 и

Графтальные растения фирмы Pixar

Метод трассирования лучей: построенное компьютером изображение бильярдных шаров

WWW.SCIAM-MAGAZINE.NAROD.RU

3 обозначают соответственно [и]. Если *i*-й символ цепочки *stringA* оказывается равным 0, то программа подставляет последовательность символов 1,2,0,3,1,2,0,3,0 в девять прилегающих друг к другу позиций массива *stringB*, начиная от индекса *j* (первая еще не заполненная позиция в массиве *stringB*). Единственный цикл в первой части программы PLANT содержит четыре таких условных оператора, по одному на каждый из возможных символов. Индекс *j* в цикле играет роль указателя на тот символ в текущем поколении, который обрабатывается в данный момент. Тело этого цикла выполняется для стольких поколений, сколько потребуется художнику-графталисту для воплощения его творческих замыслов. На каждой стадии программа PLANT может запрашивать пользователя, нужна ли еще одна (еще более длинная) цепочка символов.

Во второй, графической части программы PLANT цепочка символов, полученная в результате выполнения первой части программы, превращается в рисунок. Процедура отображения рекурсивна. Пока не встретится левая скобка, или 2, программа рисует последовательность линейных отрезков в заданном направлении. Когда встречается левая скобка, следующий отрезок прямой проводится в новом направлении, под углом 45° (против часовой стрелки) к прежнему. Эта подпроцедура завершается при появлении соответствующей правой скобки. Здесь можно нарисовать листик (его форма и цвет выбираются произвольно). Появление еще одной левой скобки означает повторное выполнение подпроцедуры, с той лишь разницей, что теперь направление меняется на угол 45° по часовой стрелке.

В программе PLANT используется множитель масштаба, величина которого зависит от сложности изображаемого растения. Например, растение в *n*-м поколении имеет высоту порядка 2^n отрезков. Если на высоте экрана укладываются 200 пикселов, то элементарные отрезки должны быть короче $200/2^n$. Изобретательные читатели, несомненно, поэкспериментируют с различными вариантами генерирующих грамматик, углов ветвления и формы листиков. В результате этих вариаций на экране появятся пейзажи из различных трав, кустарников и деревьев (хотя, наверное, и не очень реалистичных).

Хрустальный шар в гипотетическом фильме Толкейна выполняется по методу трассирования лучей, пламя на стенах крепости имитируется путем прослеживания за движением большой совокупности мелких частиц.

Метод трассирования лучей требует задания трехмерной геометрии изображаемой сцены и положения источника света. Покидая источник, каждый луч света следует по сложной траектории, многократно отражаясь и преломляясь. Глаз наблюдателя, который смотрит на сцену, перехватит лишь некоторые лучи и упустит многие другие. Чтобы сэкономить машинное время и память, трассировка лучей осуществляется в обратном направлении. Допустим, что лучи исходят из глаза наблюдателя. Широкий пучок лучей распространяется по сцене. Если луч встречает на своем пути отражающую поверхность, его направление меняется в соответствии с законами оптики. В конце концов луч попадет на поглощающую поверхность и примет тот цвет, который по условию соответствует данной поверхности. Этот цвет присваивается пикселу, который соответствует направлению исходного луча.

Пример изображения, получаемого с использованием метода трассирования лучей, показан на нижнем рисунке на с. 107. В сцене с бильярдными шарами лучи описывают сравнительно простые траектории. Тем не менее на поверхности шаров мы можем видеть отражение бильярдного зала и человека с кием в руках, наблюдающего за движением шаров после очередного удара.

Большая система частиц, которой можно воспользоваться для воспроизведения пламени на стенках крепости, является как бы логическим продолжением тех маленьких облачков из светящихся точек, которые символизировали миниатюрные взрывы в видео- и компьютерных играх. Однако система частиц, разработанная фирмой Pixar, намного сложнее. В определенной области экрана в непрерывном движении живет стая крошечных частиц. Под управлением программы каждая точечная частичка движется в соответствии со строгими динамическими законами. Она рождается, некоторое время живет и двигается, может даже произвести на свет несколько новых частиц и наконец умирает.

Возможности подобной системы частиц были с блеском продемонстрированы в сцене из фильма «Звездный марш II — Гнев хана». На мертвую планету, поверхность которой усеяна кратерами, падает бомба «генезиса». От ее взрыва распространяется причудливо искрящееся кольцо пламени, в конце концов охватывающего всю планету. Когда пламя утихает, мы видим, что поверхность планеты чудесным образом преобразовалась — теперь она покрыта цветущей роскошной биосферой. Этот эффект был продемонстрирован специалистами фирмы Pixar в те времена, когда она была еще лабораторией компьютерной графики под названием Lucasfilm. Распространяющееся огненное кольцо состояло из ансамблей частиц, в которых отдельные частицы иногда порождали новые ансамбли. Частицы этих новых ансамблей устремлялись высоко вверх над поверхностью планеты, изменяли свой цвет и затем падали вниз под действием гравитационных сил.

Э. Смит, возглавляющий отдел исследований и разработок фирмы Pixar, провел меня по ее лабораториям во время моего визита в Сан-

Кадр из фильма «Звездный марш II»

WWW.SCIAM-MAGAZINE.NAROD.RU

Рафаэль. Смит уже знаком читателям нашего журнала по своим работам в области клеточных автоматов (см. рубрику «Занимательный компьютер»: «В мире науки», 1985, № 10). Кроме Смита, первым применившим графтальный подход к «компьютерной растительности», я познакомился с Л. Карпентером, специализирующимся на построении изображений гор с помощью фракталий, Р. Куком, экспертом по методу трассирования лучей, и У. Ривсом, изобретателем ансамблей частиц. В разгаре дискуссии о путях развития графического программного обеспечения Смит немало удивил меня, сказав, что главная цель его фирмы состоит не столько в создании специальных эффектов для голливудских фильмов, сколько в разработке специализированного графического компьютера Pixar Image Computer.

Основу графического компьютера фирмы Pixar составляет оперативная память объемом 24 мегабайт для хранения массива 2000 × 2000 элементов изображения. Этим обеспечивается более чем достаточное разрешение для большинства приложений. Каждый элемент изображения (пиксел) представлен 48 битами памяти, что позволяет хранить обширную информацию, скажем, о цвете и прозрачности изображения. Огромная память компьютера Pixar будет управляться четырьмя работающими параллельно высокопроизводительными полностью программируемыми процессами. Они в состоянии выполнять около 40 млн. операций в секунду, обеспечивая быстродействие на несколько порядков выше по сравнению с быстродействием обычных компьютеров. Блок отображения графической информации взаимодействует с памятью компьютера со скоростью 480 мегабайт в секунду.

Первые образцы нового компьютера фирмы Pixar были выпущены в мае прошлого года. Они предназначены для работы в комплексе с медицинскими приборами, с дистанционными датчиками, со средствами инженерного проектирования, а также для создания мультипликационных фильмов. Может быть, именно с помощью таких компьютеров и будет создан мой гипотетический фильм.

В СТАТЬЕ, посвященной программам, генерирующим узоры для обоев, которая была опубликована в ноябрьском за 1986 г. номере нашего журнала, речь шла о почти повторяющейся структуре рисунков, создаваемых компьютером и весьма подходящих для обоев. Однако на самом деле изображения подобного рода были известны еще предыдущему поколению читателей. М. Россману, журналисту и политическому комментатору из Беркли (шт. Калифорния) принадлежит фраза «обои для ума». Он первым применил ее в 1971 г., описывая деликатные узоры — галлюцинации, которые при закрытых глазах наблюдали люди, принявшие дозу препарата ЛСД. Россман писал: «Это было похоже на обои, узор которых никогда не выглядел полосатым, но был преимущественно повторяющимся: здесь были упорядоченные «созвездия» попугаев, морских звезд, электрических лампочек и снежных хлопьев, сложнейшие неописуемые образы, подобные пятнам, используемым в тесте Роршаха*, причудливо сменяющие друг друга, эволюционирующие один из другого, как в бесконечной цепи метаморфоз, оживленных как в мультипликационном фильме».

Читатели, наверное, помнят, что самая простая программа, создающая изображения такого рода обоев, принадлежала Дж. Коннету из Миннесотского университета. Согласно его «рецепту», координаты каждого пиксела должны возводиться в квадрат и складываться между собой. Результат усекается до ближайшего целого числа. Если полученное целое число четно, пиксел будет окрашен в черный цвет, если же оно нечетно, пиксел будет белым. Сходство создаваемого узора с узорами на обоях заключается в наличии горизонтальных и вертикальных повторений и муаровой фактуры рисунка, достигаемой наложением друг на друга двух неявно выраженных узоров. Первый из этих узоров представляет собой прямоугольную решетку пикселов, соответствующую геометрии экрана. Второй узор — это семейство концентрических колец из тех точек на плоскости, для которых процедура Коннета дает нечетные числа. Подобный рисунку на обоях этот узор возникает в результате повторяющейся интерференции его элементов в горизонтальном и вертикальном направлениях. Несколько последовательно расположенных точек решетки может попасть во внутреннюю область кольца, следующие несколько точек — во внешнюю и т. д. По мере того как растет расстояние от центра, кольца становятся все тоньше, попадания и промахи наблюдаются вдоль любой линии, состоящей из точек решетки.

П. Брауну из Сими-Вэлли (шт. Калифорния) этот процесс показался невыносимо медленным. Экран его компьютера заполнялся так долго, что он решил ограничиться лишь частичной выборкой из полной картинки и высвечивать лишь каждый девятый пиксел как по горизонтали, так и по вертикали. В этой сжатой форме изображение выглядело совершенно непохожим на исходное. На самом деле Браун изменил размер решетки, в результате чего изменился характер интерференции узоров.

О. Смит из Порт-Таунсенда (шт. Вашингтон) пришел к выводу, что можно значительно изменить рисунок муарового узора, просто изменив закон, по которому определяется цвет точки. Как и многие другие читатели, Смит экспериментировал с формулами, отличными от формулы Коннета. Например, некоторые суммы или произведения простых тригонометрических функций порождают буйные вихревые узоры, напоминающие роскошно разукрашенные внутренние стороны обложки дорогих книг в твердом высококачественном переплете. Смит отмечает также, что подобные муаровые узоры и другие изображения, порождаемые интерференционными эффектами, часто возникают в компьютерной графике. Причем в большинстве случаев эти изображения возникают как побочные, нежелательные эффекты при отладке графических программ со сложными регулярными вариациями изображения.

Я был очень тронут миниатюрным воспроизводимым вручную мультфильмом, присланным Д. Рэймондом из Оринды (шт. Калифорния). Фильм состоит из подобранной последовательности крошечных отпечатков обоев Коннета с постепенно увеличивающимся размером решетки. Отпечатки плотно сшиты между собой в виде маленькой книжечки, и если быстро перелистывать страницы, держа их большим пальцем, то можно наблюдать короткий мультипликационный фильм.

* Герман Роршах — швейцарский врач-психиатр, скончавшийся в 1922 г. Его именем назван один из тестов на умственное развитие и характер человека. Тест основан на том, как человек интерпретирует чернильные пятна в категориях, позволяющих понять его эмоциональные и интеллектуальные особенности. — *Прим. перев.*